DIVERSE SEQUENCE SEARCH AND ALIGNMENT

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Elif Eser

August, 2013

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Hakan Ferhatosmanoğlu(Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Tolga Can

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Öznur Taştan

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

DIVERSE SEQUENCE SEARCH AND ALIGNMENT

Elif Eser
M.S. in Computer Engineering
Supervisor: Assoc. Prof. Dr. Hakan Ferhatosmanoğlu
August, 2013

Sequence similarity tools, such as BLAST, seek sequences from a database most similar to a query. They return results significantly similar to the query sequence that are typically also highly similar to each other. Most sequence analysis tasks in bioinformatics require an exploratory approach where the initial results guide the user to new searches. However, diversity has not been considered as an integral component of sequence search tools yet. Repetitions in the result can be avoided by introducing non-redundancy during database construction; however, it is not feasible to dynamically set a level of non-redundancy tailored to a query sequence. We introduce the problem of diverse search and browsing in sequence databases that produces non-redundant results optimized for any given query. We define diversity measures for sequences, and propose methods to obtain diverse results extracted from current sequence similarity search tools. We propose a new measure to evaluate the diversity of a set of sequences that is returned as a result of a similarity query. We evaluate the effectiveness of the proposed methods in post-processing PSI-BLAST results. We also assess the functional diversity of the returned results based on available Gene Ontology annotations. Our experiments show that the proposed methods are able to achieve more diverse yet similar result sets compared to static non-redundancy approaches. In both sequence based and functional diversity evaluation, the proposed diversification methods outperform original BLAST results significantly. We built an online diverse sequence search tool Div-BLAST that supports queries using BLAST web services. It re-ranks the results diversely according to given parameters.

Keywords: diversity search, sequence alignment, data analysis.

ÖZET

SEKANS ARAMADA ÇEŞİTLİLİK VE HİZALAMA

Elif Eser
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Assoc. Prof. Dr. Hakan Ferhatosmanoğlu
Ağustos, 2013

BLAST gibi sekans arama araçları, bir sorgu sekansı için, seçilen veritabanındaki en benzer sonuçları bulmayı amaçlar. Sorguya benzer sonuçlar, kendi içinde de benzerlik göstermektedir. Biyoenformatikteki bir çok analiz yeni aramalar için daha geniş bir yaklaşım gerektirir ve ilk sıralardaki sonuçların daha farklı çeşitler sunarak yol gösterici olması beklenir. Fakat, şu anki arama sistemlerinde çeşitlilik henüz tamamlayıcı bir parça olarak sunulmamaktadır. Tekrar eden sonuçların azaltılması adına, sekans veritabanları oluşturulurken belli bir gereklilik seviyesine bakılmaktadır. Ama, bu durum dinamik olarak oluşturulmuş sonuç kümelerinin gereklilik seviyelerini kontrol etmek için uygun değildir. Bu tezde, öncelikle, sekans araması için çeşitlilik arama problemi üzerinde durduk. Tüm sorgular ve sonuçlar için kullanılabilecek çözümler geliştirmeye çalıştık. Sekans arama araçlarında alınan sonuçlara uygulanabilecek, olası çeşitlilik ölçekleri geliştirdik. Bunların yanı sıra, deneyleri değerlendirmek için de objektif bir değerlendirme ölçeği tanımladık. Çeşitlilik algoritmalarının etkinliğini PSI-BLAST aracı kullanılarak alınmış sonuçlar üzerinde değerlendirdik. Ayrıca, sonuçların biyolojik açıdan anlamlı olup olmadığını kontrol etmek için gen ontolojilerinin kullanıldığı bir fonksiyonel çeşitlilik ölçeği belirledik. Yapılan deneyler, önerdiğimiz metotların orijinal arama sonuçlarından, hem fonksiyonel hem sekans tabanlı analizlerde, istatistiksel olarak daha üstün olduğunu gösterdi. Bunların dışında, geliştirdiğimiz yöntemlerin kullanımını sağlamak için BLAST web servislerini kullanan Div-BLAST adında bir web arama aracı geliştirdik. Bahsi geçen araç öncelikle verilen paramatreleri kullanarak BLAST üzerinde arama yapmakta; daha sonra bu aramada elde edilen sonuçları çeşitlilik unsurunu hesaba katarak yeniden sıralamakta ve BLAST kullanıcılarının alıştığı bir arayüze benzer şekilde sonuçları sunmaktadır.

Anahtar sözcükler: çeşitlilik arama, sekans hizalama, veri analizi.

Acknowledgement

I am thankful to the Scientific and Technological Research Council of Turkey (TUBITAK) and Turkish Academy of Sciences (TUBA). During my master, I have been supported with BIDEB program of TUBITAK.

I would like to express my deepest appreciation to all those who provided me write my thesis. Especially, I would like to thank my supervisor Hakan Ferhatosmanoğlu. I appreciate all his guidance, assistance, motivation and encouragement. His contributions of time, ideas, and advices make my M.S. experience very productive and stimulating. He is always an inspiring professor and mentor for many issues I have consulted.

I am grateful to Tolga Can who always helps with his ideas, works and feedbacks whenever I ask. He is always supportive in the project of my thesis. I would also like to thank to Öznur Taştan who is a helpful and kind professor with her ideas and attitudes. I am really glad them to be in my thesis jury.

I also appreciate my mother, Sübhan, father, Binali, and sisters, Esra and Esma, because they have been always on my side whatever I decided to do. They are always supportive and kind in any issue.

Last but not least, I want to thank my friends. Foremost, I am very thankful to Seher Acer, Esra Akbaş, Gökçen Çimen, Bengü Kevinç, Gizem Mısırlı, and Zeynep Korkmaz. Although we have met for two years, I owe many things to them. I also express my gratefulness to Aslıhan Akın, Büşra Altınsoy, Merve Baş, Kerim Çolak, Betül Demirkaya, İhsan Karataş, Dudu Gülcan Kırman, Mehmet Toker, Ayşe Nur Toptaş, and Gözde Çetin Uzun. For many years, they have been always supportive and encouraging to me in any way. Also, I am thankful to Shatlyk Ashyralyyev, İzzeddin Gür, Mehmet Güvercin, Mustafa Korkmaz, Caner Mercan, Gülden Olgun, Nermin Samet, Fadime Şener, and Can Telkenaroğlu. Without my friends, many things would be more difficult to struggle.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Sequence similarity search is one of the earliest and most commonly employed tools of bioinformatics by molecular biologists. In the current sequence search tools, the results retrieved from the database are typically also highly similar to each other. For many bioinformatics tasks, the result set needs to be diversified to produce a subset of results containing sequences well aligned with the query but sufficiently different from each other. This need is apparent in the use of non-redundant databases such as the *nr* database used in BLAST. However, to the best of our knowledge, no sequence similarity search tool incorporates diversity to the search algorithm. Search diversification has been studied in information retrieval, but it has not attracted attention in bioinformatics yet.

Sequence similarity search is an area that would benefit to have more diverse results instead of just top similar results. Identification of all functional domains of a query sequence, which may be comprised of separate homologous domains in different sequences, can only be established by an approach whose main purpose is to cover most of the query sequence other than finding the most similar sequence. Here an example is provided to explain diversity for sequences. In this case, 7 sequences are returned as results for a given query and top 4 diverse ones are demanded. The result set is comprised of the aligned parts of results with respect to the query. In the instance, the aligned regions of the sequences are bold and red.

The query: ATGTCCATCGTTTAA

The result set from a local alignment tool:

1. **ATGTCCATCGTTTAA**
2. **ATGT**AAC**TCGTTTAA**
3. **ATG**CAAC**TCGTTTAA**
4. **A–GT**AAAC**CGTTTAA**
5. GCTA**CCATCGTTTAA**
6. GCTAG**CATCGTTTAA**
7. **ATGTCCATCGT**GTAC

Diversified 4 results are below:

1. **ATGTCCATCGTTTAA**
2. **ATGT**AAC**TCGTTTAA**
3. GCTA**CCATCGTTTAA**
4. **ATGTCCATCGT**GTAC

In diversified result set the third and fourth result sequences are omitted because of the similarity to the second sequence. It means their alignments have the same characteristic to one of the chosen sequences. Then, the fifth one is chosen which means the sixth sequence is excluded from the new result set due to the fact that its alignment for the query is highly parallel to the alignment of the previous sequence. The example is given for visualizing diversity of sequences. In this thesis, we formalize the problem of diversification and investigate methods to post-process results from the commonly employed search tools to remove redundancy from the results and enable an exploratory browsing. An example to such searches is to find proteins each with different functions but similar enough to the query sequence. Different segments of the primary structure may correspond to different functional domains. Tools such as BLAST incorporate a domain identification step and present the identified domains to the user in addition to the query results. However, domain identification is limited to known, characterized domains and novel domains in the query sequence will be overlooked by this approach. Such novel domains may be shared by some of the database sequences and a diverse search may identify these regions. For this purpose, finding a diverse set of regions with similar segments would be a more appropriate approach than simply investigating the top similar sequences. With our proposed method,

we are also able to control the effect of diversification, based on the dissimilarity of biological functions of sequences.

Sequence alignment is utilized to arrange the sequences of DNA, RNA, or amino acid sequences to detect the regions of similarity. Global alignment follows a general similarity measure and attempts to align every residue in every sequence using gaps, and local alignment focuses on determining similar subregions. Sequence search tools such as BLAST [1] [2] and FASTA [3] seek similar sequences to a given query in large sequence databases. Our proposed approach is applicable to post-process the results of any sequence similarity search tool. However, for the experiments, we focus on Position-Specific Iterated BLAST (PSI-BLAST) which seeks locally similar sequences on protein databases by using profiles.

Although diversity search is not explicitly investigated yet in the context of browsing sequence databases, one can decrease the redundancy level of these databases by a preprocessing procedure. Commonly used protein sequence databases such as UniProtKB, UniProtKB/Swiss-Prot, UniParc, and UniRef databases have reduced, non-redundant versions. UniProtKB includes two different databases: UniProt/TrEMBL and UniProt/Swiss-Prot. In UniProt/TrEMBL database, for the fully identical, full-length sequences from one species there is one record. UniProt/Swiss-Prot is built with different representative sequences for sequences encoded by one gene in one species. UniParc and UniRef databases comprise also of representatives for 100 percent identical sequences, regardless of the species. Additionally in the UniRef databases, subfragments are also included as different records apart from full-length sequences. These databases implicitly remove the same alignment results by eliminating identical sequences or fragments from the databases. This preprocessing is done in design time and is independent of the query sequence. While queries can avoid identical sequences in the results, most still contain results with too much redundancy, as we illustrate also in the experimental section.

We adopt novelty model as diversification approach. In our model, we *implicitly* aim to find novel sequences which are aligned with different sections of query from those are already covered by the current result set. The word *implicitly*

refers to which we expect to recognize the sequences with novel regions to query by comparing the results with each other, not to the query [4]. It means that if the results are different enough from the others, we cover all possible regions of query, and eventually we obtain a global diversity on the result set. We present two methods, BitDiversity and EntropyDiversity, which iteratively construct a set of results that are diversely aligned with the query sequence.

We built an online diverse sequence search tool named as Div-BLAST that supports queries using BLAST web services. The tool renew the order of given result set according to given parameters. Apart from BLAST search parameters such as database, program, query etc., Div-BLAST makes users to choose one of aforementioned diversity algorithm and diversity rate. Although our diversification methods do not need any parameter from outside, we add a feature for allowing users to observe similarity and diversity tradeoff. User may utilize the rate feature even after getting results of search. Div-BLAST recorded the old queries with a unique id, gives permission to download the result set, makes users to be able to arrange the results in ascending or descending order with respect to score, e-value, coverage, etc.

We propose a novel diversity measure based on Rao's quadratic entropy to evaluate the quality of results. Moreover, we evaluate the diversity of the protein functions using a molecular functional ontology subset of the Gene Ontology (GO) terms. For each evaluation, we also test the significance of results with Wilcoxon signed-rank test which is a non-parametric statistical hypothesis test for two sets of samples. We compare the results of both diversity methods with original BLAST results. Additionally, we give the query coverage comparison results of diversified sets and the original set. Since, one of the aims of diversification is to find diverse regions of queries, the diversified set achieves a complete coverage more rapidly than the original set. We test the significance of coverage results with Wilcoxon test, as well.

Chapter 2

Background

2.1 Sequence Alignment

Sequence alignment is the most common way to explore the similarity between sequences. It is based on arranging the sequences belonging to DNA, RNA or proteins for finding similar that may be caused by relationships between them such as evolutionary, structure-based or functional [5]. Basically, there are different methods for pairwise sequence alignment: global and local. Global alignment aims to align every residue in both sequences by forcing them to have an equal size by using gaps. Its purpose is to have a general similarity value over whole sequences. On the other hand, the objective of local alignment is to determine similar regions; it does not care the global order [6] [7]. The local and global alignment example for the same sequences is presented in Figure 2.1. As seen in the figure, local alignment tries to identify a similarity and it may not be aligned all residues of both sequences while global alignment considers the whole pieces of both of the sequences.

The most common algorithm for global and local alignment are Needleman-Wunsch and Smith-Waterman, respectively. Needleman-Wunsch algorithm is designed in 1970 [6]. The algorithm is basically based on dynamic programming which attempts to find the solution of the subproblems instead of solving the

```
Global FTFTALILLAVAV
       F--TAL-LLA-AV

Local  FTFTALILL-AVAV
       --FTAL-LLAAV--
```

Figure 2.1: An illustrative example for the difference between local and global alignment.
Available at: http://en.wikipedia.org/wiki/Sequence_alignment

whole problem completely at the same time. It needs a similarity matrix which defines the similarity scores between each possible letter included in sequences. For amino acids, BLOSUM matrices are commonly employed as substitution matrix especially BLOSUM62 with 62% similarity threshold. These similarity, or substitution matrices were built by examining a database named Blocks comprised of aligned segments of homologous proteins [8]. The scores in the matrices are derived from alignments of the homologous sequences by looking at the frequency of any amino acid pair, simply. In global alignment algorithm, a gap penalty, i.e., a negative similarity score, for the pairs including gaps also exists. Eventually, the alignment that maximizes total similarity score including gaps, it will be chosen as the final alignment.

Smith-Waterman algorithm is proposed in 1981 [7] by using the same idea in the previous algorithm. The difference is that there is no penalty for every gap. If an aligned region is started the gap is penalized, however if the gap is not included by an alignment, it would not affect the total score.

2.1.1 BLAST (Basic Local Alignment Search Tool)

BLAST is a very popular tool searching similarity on primary structure of biological sequences such as amino acid or nucleotide sequences. It is first introduced in 1990 [1]. It is also a local alignment algorithm that has a different methodology from Smith-Waterman. Basically, BLAST algorithm extracts k-letter words from query, scans database over the word list, and tries to extend the matches

until finding optimal high-scoring segment pairs (HSPs) for given query. Although Smith-Waterman gives the best result for an alignment and BLAST is a heuristic method, BLAST outperforms Smith-Waterman in terms of speed [9]. Here is a tradeoff between being sensitivity and speed; however especially in large databases, the latter gains more importance. BLAST compares nucleotide or protein sequences to large sequence databases, calculates the statistical significance of matches and returns the results with attributes such as query coverage, total score, max score, e-value, and maximal identity. It could be said that these parameters are correlated in some way. For instance, there is a negative correlation between score and e-value. Score refers to the score of high scoring pairs (HSPs) of the alignment and e-value, expect value, is a statistical significance parameter related to the hits number expected to be seen by chance. Lower e-values indicate more significance of results. PSI-BLAST [2] is more sensitive than the original version of BLAST using pairwise comparisons between sequences. This is because profiles are built by considering evolutionary relationships and using them enables detection of distant relatives of a protein. As diversification may be applied for *blastp*, original BLAST for protein-protein search, it could be possible to use for all versions of BLAST including nucleotide-nucleotide BLAST, *blastn*, and translation BLAST types such as *blastx, tblastx, tblastn*. The translation models may compare nucleotides to amino acids or vice versa.

In addition to different versions of algorithms, there are more than one way to utilize BLAST such as online, stand-alone or via web services. Online version supplies an interface to search queries and shows the results with their specifications on web. Stand-alone version makes BLAST software run on a local computer without using internet connection. With the help of web services, one can employ BLAST utilities programmatically in many languages such as Java, C# etc.

2.2 Diversity

Diversification aims to produce results that are similar to the query but dissimilar to each other, basically. Although there is no prior work on diversification in sequence searching, the notions of diversity and novelty are present in the context of information retrieval and recommendation systems. The diversity problem is known NP-hard to optimize; therefore, most algorithms presented in diversification studies have greedy approaches that choose samples from a given result set by iteratively selecting the local optimum for the current set. The main purpose is to have the maximum coverage with the diversified result set to the given query. The coverage may be provided *implicitly* or *explicitly* [4]. Seeking for coverage implicitly refers to expect to have the maximum coverage of all aspects of query without checking the query. In the approach, it is assumed that it will be able to obtain full coverage and prevent overall redundancy by providing maximum difference among the samples in the result set.

Carbonell and Goldstein [10] was the first to introduce the Maximal Marginal Relevance (MMR) for text retrieval and summarization. MMR builds a result set by maximizing the query relevance and minimizing the similarity between documents in the result set. The method uses a parameter (λ) that specifies the proportions of relevancy and novelty. Although it has a simple approach to optimize diversity problem, this study has guided many works related to diversity.

Jain *et al.* [11] [12] propose two greedy solutions for the k-nearest diverse neighbor search for spatial data. In both of the approaches, R-tree index is employed while optimizing relevance and diversity. There are two notions in these studies: Immediate Greedy (IG) and Buffered Greedy (BG). IG incrementally populates the result set $\mathcal{R}$ with the nearest result points only if they provide diversity enough to the points already included in $\mathcal{R}$. BG is a kind of developed version of IG; it attempts to reduce the negative impacts of the previous. In the algorithm, before a data point is accepted as one of $\mathcal{R}$, its effects to $\mathcal{R}$ are observed during a number of iterations after and if it diversifies $\mathcal{R}$ enough, it will be added to $\mathcal{R}$. Note that they use the R-tree index only for finding the nearest neighbors of the query among the points of whole data set.

Chen and Karger [13] propose a probabilistic model to maximize diversity by assigning negative feedbacks for the retrieved documents that are already included in current result set. They do not just penalize the irrelevant documents to the query, also the relevant and observed ones.

Yu [14] [15] investigate the diversification issue in recommendation systems with two heuristic algorithms, *Swap* and *Greedy*, to maximize the diversity by taking into account different relevance constraints. They also indicate diversification is related to find a balance between relevance and novelty. Swap algorithm swaps items by starting with top-k relevant ones by excluding the items which are less likely to make contribution to the set in terms of diversity. Greedy algorithm populates the diversified set with the most satisfying item which is relevant and distant enough in every iteration.

Liu and Jagadish [16] diversify the results by adopting the approach of clustering for the Many-Answers Problem. They propose a tree-based approach to choose one representative from each cluster consisting of diverse results. A tree-based approach is adopted as clustering method to obtain efficiency while finding representatives.

The works presented above all have an implicit manner in diversification. There are also studies with an explicit diversification methodologies. This kind of approaches aim to implement algorithms considering the taxonomy of both queries and documents to build a diversified set such as the studies of Vee [17] or Clarke [18]. Vee *et al.* has worked on diversification over a structured database for online shopping consisting of the objects denoted with a set of features. The goal of their diversification system is to serve a result set containing a set of items that are as diverse as possible according to features. According to their approach it is not possible to supply full satisfaction on all features by post-processing over a result set of search; the diversification should be applied during searching. As implied, they focuses on novelty rather than relevance to the query.

The study of Clarke *et al.* [18] is related to diversification in answering questions context. They attempt to solve the problems of ambiguity in queries and

redundancy in retrieved documents. They develop an evaluation framework taking into account both novelty and diversity, i.e., novelty and relevance together. Questions and answers comprise of "information nuggets" which are defined as an atomic piece of information about text [19], and relevance is based on a function of the nuggets contained in both the questions and the answers. The work of Agrawal [20] is a similar study to Clarke's with the difference that they also consider the relative importance between nuggets and the possibility that different documents with the same nugget may serve different extent to the users.

Chapter 3

Problem Definition and Methods

3.1 Problem Explanation

We aim to find k diverse sequences from the result set of a query searched using a sequence search tool, e.g, PSI-BLAST. Note that k is a user tunable parameter which is optional because our algorithms are not based on the value; they are incremental. In other words, the first k diversified results are the same as those in the diversified set with k=k+*1*. The algorithms may run as re-ranking the result set regarding diversity. The k parameter provides speed without waiting all sequences to be ranked. We expect the k diverse sequences to have alignments with the query that are different from each other. In other words, we want to choose k novel results which have query coverage on different sections of the given query or novel residues within the same alignment region. We present methods for systematizing the diversification problem. In accordance with the above-mentioned diversity definition, in our approaches, we deal with not full-length result sequences but the aligned fragments with the query. In the rest of the thesis, the term result sequence refers to an aligned fragment.

Equation 3.1 represents the general formulation of diversity for our approaches, namely BitDiversity and EntropyDiversity. Both of these approaches

are iterative, i.e., in each iteration the sequence which provides maximum difference is added into the current diverse set regardless of the original order in the result set. We initialize the diverse set with the first sequence of the original result set. We fix the length of all result sequences to that of query enlarged with the gaps formed in the alignments of query and any result sequence. The algorithm stops when the size of the current diverse set reaches k. The proposed BitDiversity and EntropyDiversity will be detailed in the following sections. Briefly, BitDiversity is based on the average of the differences between candidate and each result sequence whereas in EntropyDiversity it is the general entropy of result sequences and candidates together. The proposed algorithms are executed as a post-processing of the search results which involve aligned sections of result sequences.

$$diversity = \underset{D_i \in R/S \;\; i \leq k}{\operatorname{argmax}} [difference(D_i, Q, R')] \tag{3.1}$$

Here, $\mathcal{D}_i$ is a result sequence included by diversified set $\mathcal{R}$ which is a subset of all result sequences $\mathcal{S}$. The size of $\mathcal{R}$ depends on k. $\mathcal{R}'$ is the chosen diverse set before $\mathcal{D}_i$. Q represents the query which is used in *difference* formula characterized by diversification methods.

To exemplify the problem, we provide a result set for a given sequence by using BLAST. In the example, the program returns 27 different sequences as seen in Figure 3.1. In the result set, there are just aligned parts of result sequences with respect to the query. In addition to the information, we also know which sections of result sequences are included for the alignment with the query. Figure 3.1 illustrates the top-4 diverse results our approach returns, which are underlined in red. The parts of the query aligned with the diverse result sequences are also seen in the figure. The example illustrates our pairwise bit comparison approach, which is the simpler of the two proposed diversification approaches. Initially, the diversification algorithm starts with the first sequence in the original result set. As the second element of the set, the last sequence is chosen, which is the most distant sequence to the current set (with the first full coverage result). The second sequence in the BLAST results is selected as the third element; because,

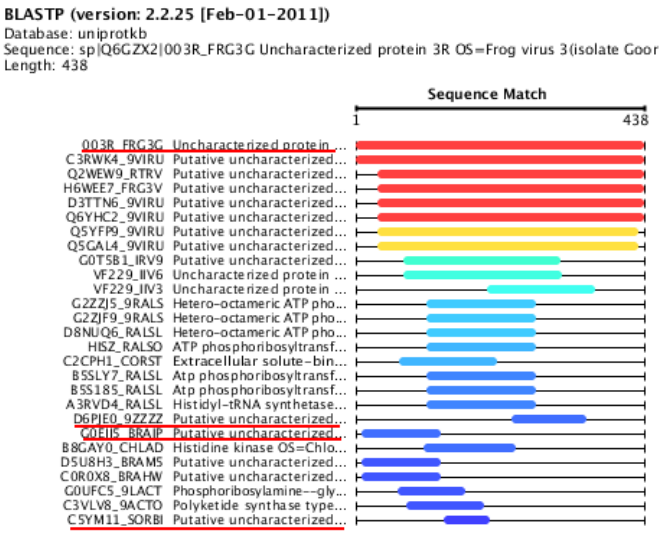


Figure 3.1: An example from BLAST. Underlined sequences are chosen as top-4 diverse results.

it has no intersection with the second elements and has the least intersection with the first one due to its length. Lastly, the sequence named as G0EIIS_BRAIP is inserted in the diversified set due to no intersection with the second and third sequences in the current set.

3.2 Pairwise Bit Comparison

Algorithm 1 presents our greedy heuristic that selects a sequence from the initial result set $\mathcal{S}$ in each iteration, and constructs the diverse k results after k iterations. In every iteration, the algorithm scans the whole unselected result list. In a sub-iteration, there is also a loop that finds the difference between the candidate result and each sequence which is in the current diversified result set. In the approach, BitDiversity, sequences are treated simply like bit sequences. The aligned residues of a result sequence with respect to query are marked as 1, otherwise it is 0.

It means every sequence is also represented as a d-dimensional binary vector that has 1 or 0 referring to matched and unmatched residues. BitDiversity uses

the bit sequences for calculating the difference of two sequences. Here the difference is computed with the division of the total number of different bit residues in the alignment by the count of the bits of total covered region. The nominator is calculated with the XOR operation which is a bitwise operator that makes the result bit 0 if a matching occurs on the other hand the result bit is 1; and the denominator with the OR operation that gives 1 as result unless both of the bits are 0 (3.2):

$$dif(M, L, Q) = \frac{\sum_{j=1}^{l} bits(M, Q)_j \oplus bits(L, Q)_j}{\sum_{j=1}^{l} bits(M, Q)_j \vee bits(L, Q)_j} \tag{3.2}$$

where l is the length of sequence, *bits* converts result sequences, M and L, to bitwise sequences with respect to the query Q. The formula divides the aggregation of the XOR results for each position j in M and L by that of the OR results.

Basically, the total number of 1s, after the XOR operation represents the difference, the total of substractions, of given two sequences; and, the number obtained after OR operation indicates the number of union of the sequences. The main objective of the division instead of just using the difference is to provide fairness between especially long-long and short-short sequence pairs. For instance, without the division, it would give the same diversity measure when the same amount of different residues between the pairs of short and long sequences occurs; even if the long sequences are almost overlapped and the short ones almost in different locations. Apart from the measure of two sequences, diversity between a sequence and a set of sequences can be defined with various patterns, such as the linkage computations [21]. In single and complete linkage approaches, the diversity-relevance measure between a sequence s and a sequence set $\mathcal{R}$ depends on the difference between the sequence and the most similar (single linkage) or most different (complete linkage) sequence from the sequence set. The minimum or maximum pairwise difference between s and the sequence of $\mathcal{R}$ specifies the diversity, depending on single or complete linkage algorithms, respectively. When the difference between s and $\mathcal{R}$ is based on the average linkage method, the average of each difference between s and each sequence of $\mathcal{R}$ is used for diversity. We experimentally observed that the average linkage approach provides the best

results.

Div function at Step 9 in Algorithm 1 depends on the diversity approach used. For BitDiversity, it calculates the diversity rate based on the average diversity rate of the current candidate sequence and each sequence in the current chosen result set.

Algorithm 1 DiversitySearch

Input: *S is the original result set, k is the length of diversified subset from initial result set and Q is the searched query*
Output: *ChosenList is the diversified subset.*

```
1: procedure DivSearch(S,k,Q)
2: Initialize m as 1 //is the counter for chosen list
3: Initialize divArr //used for diversity rates to find the greatest
4: Initialize notChosenList with S(all results except the first)
5: Initialize chosenList with { the first sequence of S }
6: while m ≤ k do
7:    set divArr{}
8:    for i = 1 → notChosenListLength do
9:       divArr[i]=Div(notChosenList[i],chosenList,Q)
10:   end for
11:   find j as the index of max valued divArr[i]
12:   add chosenList notChosenList[j]
13:   remove notChosenList[j]
14:   m++
15: end while
16: end procedure
```

3.3 Entropy Based Diversity

Entropy has been used for measuring diversity in information retrieval [22]. In the context of bioinformatics, it was applied to evaluate the quality of multiple sequence alignment, but with the opposite goal of having low entropy, i.e., to achieve a high quality alignment [23]. We follow a similar idea for sequence similarity search, where the multiple alignment of the result set is readily available in the form of a star alignment where the center sequence is the query sequence.

While the result set is similar to the query, a diverse result set implies a low scoring multiple sequence alignment. Therefore, we aim to have a high entropy score in the result for diversity. We propose an entropy based approach, EntropyDiversity, that chooses the nth sequence from the result set depending on the entropy of chosen sequences and candidate sequence together, and finds the candidate sequence that makes the entropy highest. Entropy is defined as:

$$E(R) = \sum_{j=1}^{l} - \sum_{x=1}^{s} p_{xj} * \log p_{xj} \tag{3.3}$$

where $\mathcal{R}$ is a result set, l is the length of sequence, s is the size of letter set, x represents the elements of the given alphabet, i.e., the alphabet, in other words the letter set, could be comprised of 20 amino acid letters or 0 and 1, p_{xj} is the probability of x in the j^{th} tuple of all m sequences (m is the size of result set).

In EntropyDiversity, one can look at either the entropy of amino acid residues or the bitwise entropy which deals with whether the piece of sequence is aligned. For the former, the alphabet size is 20 (possible amino acids) and in the latter it is 2 (0 and 1). We evaluate both approaches in our experimental section and decided to employ the mixture of them.

At Step 9 in Algorithm 1, we design the function *Div* is based on the combination of the amino acid and bitwise entropies by taking their average to utilize them both as presented in Equation 3.5. To balance between amino acid based and bitwise entropy, we have normalized both of them before averaging. In normalization, Equation 3.4 is used as the maximum value of the entropy. In addition to averaging two entropies, the result also is divided to the average length of the aligned fragments with the same motivation as in pairwise comparison methods to get rid of the effect of length of result sequences. Briefly, Equation 3.5 explains the *Div* function with statements.

$$\begin{aligned} ent_{max} &= -l * \sum_{x=1}^{m} p_x * \log p_x \\ &\cong -l * \sum_{x=1}^{m} \frac{\left\lceil \frac{m}{|\Sigma|} \right\rceil}{m} * \log \frac{\left\lceil \frac{m}{|\Sigma|} \right\rceil}{m} = -l * \left\lceil \frac{m}{|\Sigma|} \right\rceil * \log \frac{\left\lceil \frac{m}{|\Sigma|} \right\rceil}{m} \end{aligned} \tag{3.4}$$

where Σ is the alphabet, l is the length of the candidate sequence, and m is the

length of the multiple alignment of the result set.

$$Div(\mathcal{R}) = \frac{\textit{normalized bitwise entropy} + \textit{normalized letter based entropy}}{2 \ * \textit{average length of the sequences in } \mathcal{R}} \tag{3.5}$$

We note that for both methods, BitDiversity and EntropyDiversity, no user defined parameters are required. As we post-process the results of similarity search, the sequences in the raw result set are already similar to the query. Hence, we focus on diversification of the results.

Chapter 4

Evaluation Measures

4.1 Sequence Diversity Measure

We first propose a measure to evaluate the diversity of a sequence set that consists of result sequences already aligned with the query. We adapt a version of Raos quadratic entropy [24] [25] which is initially used for diversity of/within populations as the basis of this new measure. Quadratic entropy is used for non discrete instances; it takes into account the distances. Equation 4.1 shows the basic quadratic entropy formula:

$$E(P) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_i * p_j * d_{ij} \tag{4.1}$$

where $E(P)$ is the entropy of whole set, i.e, for all instances 1 to n, p_i represents the probability of i^{th} instance, and d_{ij} is the distance between i^{th} and j^{th} instance.

To compute entropy as in Equation 3.5, a dissimilarity matrix is needed. To convert the amino acid substitution matrices, which incorporate similarities, into dissimilarity matrices, we apply 4.2 to each element in the BLOSUM62 matrix and use it as the distance matrix for the entropy calculations. In addition to the existing rows and columns of the original BLOSUM62 matrix, we add a new row and a column for the non-aligned symbol to the query. Note that; with the new values for the matrix, we obtain a dissimilarity matrix with 0 diagonal.

$$a'_{ij} = \frac{(a_{ii} - a_{ij}) + (a_{jj} - a_{ji})}{2} \tag{4.2}$$

In Equation 4.2, a'_{ij} is the new value for the element a_{ij}. $a_{ii} - a_{ij}$ represents the raw distance between i^{th} and j^{th} element. Since a_{ii} and a_{jj} are different, the new distance values are not symmetric. To obtain a symmetric distance matrix, we use the average of the new raw distance values.

The diversity of a sequence of length l is computed as in Equation 4.3. After the result sequences are multiply aligned with respect to the query, for each tuple we calculate the quadratic entropy with the new dissimilarity matrix. The average of entropy of the tuples is the diversity rate of the given sequence set.

$$Div(P) = \frac{1}{l} \sum_{h=1}^{l} \sum_{i=1}^{s} \sum_{j=1}^{s} p_{ih} * p_{jh} * d_{ij} \tag{4.3}$$

In Equation 4.3, l is the length of sequence and s is the size of letter set including the amino acids and gap and non-aligned part symbols. p_{ih} and p_{jh} are the probability of i^{th} and j^{th} letter for h^{th} position of all m sequences (m is the size of the result set). The probability depends on the frequency on the given position and note that if the letter does not exist in the position, the probability is 0; additionally, for the same letter the entropy is also 0 since d_{ij} equals zero. Note that the dissimilarity matrix also includes the unmatched residues with respect to the query. It could be considered as a gap; however, it should be more distant from the amino acids than the gap symbol. Because, a gap is created during alignment, it is included in the alignment. Hence, the non-aligned symbol is assigned with a value twice as the value for the gap. In our experiments, this heuristic produced satisfactory results. Thanks to this process, we preserve the importance of differently aligned sections while amino acid based assessment is also taken into consideration. In other words, we are looking at the variety of matching and unmatching parts of sequences with respect to the query by considering the relationship between amino acids.

We use the above-mentioned measure to evaluate the quality of the results

returned by different diversification approaches. While the same measure can be used within the proposed diversification algorithms, we choose to follow simpler measures for reduced computation complexity, hence more efficient browsing. Performance evaluation does not have the time restrictions of an online search. Our experiments also confirm that the proposed methods did not improve even when such a complex diversity measure is used for diversification.

4.2 Functional Dissimilarity Measure

To check whether diversification methods also provide functional diversity, we propose a functional diversity measure based on Gene Ontology annotations of proteins in the result set. It has been shown that due to divergent or convergent evolution of protein functions, similar sequences may exhibit different functions [26]. In divergent evolution, the same ancestor often generates superfamilies of functional proteins catalyzing a diversity of reactions. Conversely, in convergent evolution of functional proteins, the proteins which catalyze the same reaction are independent from each other [27]. Although these conditions are valid for some of proteins, controlling functional diversity over result sets would still give insights about the importance of sequential diversity. Additionally, one of the aims of the diversity on primary structure of sequences is to obtain proteins with different functions.

Gene Ontology (GO) is an accepted concept that supplies a unification on the representation of genes and their product features in all species. Briefly, GO terms represent ontological counterparts of genes, proteins or enzymes. GO comprises three sub-ontologies: biological process, cellular component, and molecular function. The ontologies build a directed acyclic graph (DAG) whose terms are the nodes and whose edges have two kinds of semantic relations such as “is-a” and “part-of”. “is-a” is a simple class-subclass relation, where A is-a B means that A is one of the subclasses of B. “part-of” represents a partial ownership relation; C part-of D means that whenever C is present, it is always a part of D, but C does not always exist when D is seen [28].

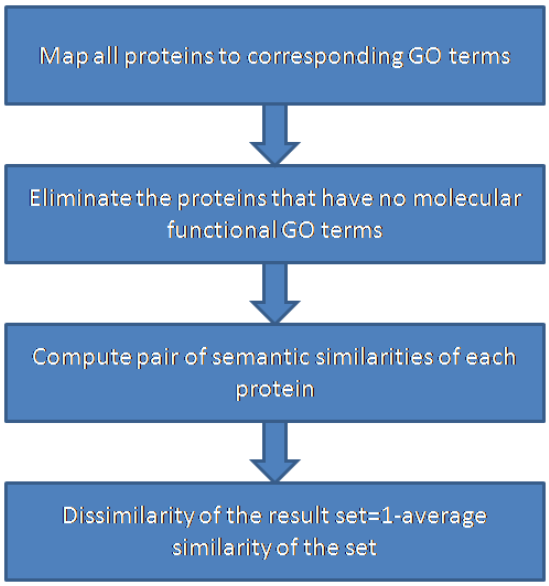


Figure 4.1: The steps of finding the functional diversity of a set of protein sequences.

To compute the functional dissimilarity of a set of protein sequences, we utilize known functions of proteins. As functional information, we use the GO terms [29] belonging to the molecular function ontology. For the ontology there are over 10000 nodes in the GO DAG. From these, approximately 890 nodes are obsolete and the others are related to other nodes with "is-a" relationship; "part-of" is not defined for functional ontological nodes.

For the similarity of functions in the molecular functional ontology, we use Wangs semantic similarity [30]. Wang *et al.* proposed a method to compute a GO terms semantics into a numeric value by aggregating the semantic contributions of their ancestor terms in the GO DAG and use the values to measure the semantic similarity of GO terms. They consider the similarity of terms not only based on their distance by using the closest ancestor, but also the specificity, i.e., depth in the DAG, of the terms. It means, according to the measure, the terms that are children of the same parent, i.e., siblings, which is close to the root of the ontology do not have the same similarity as the siblings that are close to the leaf nodes.

Figure 4.1 shows the steps of finding the dissimilarity of a result set. First

of all, the proteins in the set are mapped to their GO terms from EBI Protein-GO annotation dataset. Because the dataset is very large to mine in Java, we have partitioned the data in lexicographical order regarding protein IDs which is defined by UniProt Constitution. A protein sequence may be assigned with more than one GO terms, likewise it may not have any counterpart by the terms. All proteins except for ones which do not have any GO term annotations are included for the dissimilarity measure. In the calculation of this measure, primarily the pairwise similarities between two result sequences are computed by considering all corresponding terms referring to the proteins with Wang's similarity.After that, the dissimilarity is defined as 1-Wangs similarity, whose range is 0 to 1. The evaluation part took longer time than the other parts because both building the molecular function DAG from text data and calculating the pairwise similarities based on all terms with their ancestors are exhausting respectively.

4.3 Wilcoxon Signed-Rank Test

While assessing the results of the evaluation measures, we test the significance of results with Wilcoxon signed-rank test which is a non-parametric statistical hypothesis test for two sets of samples. In the test, first of all, the absolute values (differences within pairs) of the samples are ranked from smallest to largest. The pairs with 0 value are excluded to reduce the sample size and does not have a rank number. In ranking the list, the absolute differences with the same value have the same rank which is the average of the ranks they span. The sign of a pair depends on the sign function of the difference between 1^{st} and 2^{nd} components of given pair. All rank values belonging to the same direction e.g., negative or positive, are added up and the smaller one of the two total rank value is the test statistic, W [31]. The test return a p-value at the end of the progress which determines whether the difference between chosen N random samples from the population could be found by chance or not. The smaller p-values than a given threshold, commonly 0.05 but for more sensitivity it could be smaller, rejects the idea that the difference is not important.

We compare the results of both diversity methods with original BLAST results. For each possible k value we calculated the desired evaluation measure for both, then paired them. The pairs were given to Wilcoxon test as input. As significance threshold, we use 0.05. The significance test is done for sequence diversity, functional diversity measures and coverage rates of the sets.

Chapter 5

Experiments and Results

5.1 Dataset

We first extracted a data set by using 1000 UniRef50 [32] sequences with different lengths. The data set is used as the query set for sequence search in PSI-BLAST. UniRef (Uniprot Reference Clusters) is a non-redundant database with different threshold values: 100, 90 and 50%. Initially, UniRef100 is created to supply non-overlapping sequence sets by combining identical sequences and sequence fragments. UniRef90 and UniRef50 are built upon the UniRef100 database. Each cluster contains the sequences that have at least 90% or 50% sequence identity to the longest sequence, respectively.

5.2 Setup

We analyze similarity queries on three different databases: UniProtKB, UniProtKB/Swiss-Prot, and UniRef50. The last two databases are the commonly used non-redundant databases and the first one has also unreviewed sequences. UniProtKB is the largest protein database with 30,309,136 sequences.

The sequences may belong to UniProt/Swiss-Prot domain which includes non-redundant, manually annotated proteins or UniProtKB/TrEMBL containing the sequences that are automatically annotated but not controlled manually. UniProtKB/Swiss-Prot is also a database employed in experiment with 539,165 sequence entries. Lastly, UniRef50 consists of 21,824,511 sequences. Although UniRef50 is processed for eliminating redundancy more than Swiss-Prot, it has more entries than the other. Its reason is that UniRef50 dataset also consist of not only whole sequences, and fragments. We performed all the experiments with the *psi-blast* tool, which returns more results than the regular *blastp* because of its sensitivity.

We evaluate the two proposed diversification algorithms, BitDiversity and EntropyDiversity, by comparing with the original ordered result set of PSI-BLAST. An alternative approach would be to post-process the result ensuring each sequence to be different enough from the current set of chosen sequences. Here, the difference of a pair of sequences can be defined by their alignment score. If the results are less similar than a given threshold, say 40%, one sequence can be considered diverse enough from the other one. We observed that this alternative method did not improve the diversity of the original set as the sequence identity was computed on the whole sequences, not the query related fragments. However, we consider the query aligned diversity to find differently aligned sequences. One sequence, which may pass the threshold by taking into account the whole sequence, may not be diversified with respect to given query. Another drawback of finding a diversified list by using this baseline approach is that one may not get a result set with the desired number of sequences; it may return less number of results than expected. All the sequences passing the given non-redundancy threshold are provided in the result set.

In this thesis, all the experiments are performed on a computer with 2.27 GHz CPU and 3.0 GB RAM.

5.3 Results

The evaluation results of the proposed algorithms are illustrated in Figure 5.1, 5.2, 5.3, 5.4, 5.5, 5.6. For each possible k which is less than or equal to the result set size, we plot the average diversity rate of top-k diverse result sets of each query. For example, for the point where the k equals 35, we use the average of the diversity rates whose result set amount equals to or more than 35. We compute the average of the rates belonging to each sequence in the k point.

Additionally, we plot the diversity rates of the original PSI-BLAST result set to compare with our methods. Note that as the non-redundancy rate increases (UniRef50 > Swiss-Prot > UniProtKB), the diversity rates are getting better in original PSI-BLAST result sets (Figure 5.1, 5.3, 5.5). This is not surprising as there is a significant pre-processing in preparation of these databases. However, our methods are online and do not rely on this preprocessing, and still work considerably well even with redundant data.

5.3.1 Sequence Based Diversity

For the first evaluation, sequence based diversity using quadratic entropy, all experiments with different databases show that our results obtained with both entropic and pairwise methods have better results than the PSI-BLAST results (Figure 5.1, 5.3, 5.5). As mentioned, we test the statistical significance of our methods and PSI-BLAST with Wilcoxon signed-rank test. We use τ=**0.05** as significance threshold. In all databases, the test gives extremely small p-values ($\sim$ 0) for each k diversified result set until k$\approx$400 while comparing diversified sets and original set. The results of the proposed methods are close to each other by looking at the averages. However, we may say, significance tests show, the pairwise method is slightly more effective than the entropy based approach especially for small k (approximately 50, in UniRef50 database it is close to 100).

As expected, the diversity rates have a decreasing trend while instance number increases; as the methods try to choose the most different sequences at first. There

are minor fluctuations because of the independency of the evaluation criteria and methods diversity criteria. On the contrary, the PSI-BLAST rates are inclined to increase while the instance number decreases, since more similar results to the query are obtained at the beginning.

In addition to finding mean values for each k, we also analyzed the standard deviation related to given mean values. The results show that PSI-BLAST results have more deviation than diversity methods. The averages of the deviations are 1.4 for EntDiversity, 1.43 for BitDiversity and 1.47 for PSI-BLAST results in UniProtKB. These values in Swiss-Prot are 1.32, 1.35 and 1.38; for UniRef50 they are 0.95, 0.98, and 1.01, respectively. The difference between diversity methods and PSI-BLAST is statistically significant according to Wilcoxon test with lower than 0.05 p-value.

5.3.2 Functional Diversity

While sequential diversity is not directly correlated to functional diversity; they provide useful insights about different aspects of sequences. The functional dissimilarity rates of the result sets are illustrated in Figure 5.2, 5.4, 5.6. In the graphs, the maximum instance number is less than the original result set size because not all sequences are annotated with gene ontology terms and, we do not compute functional dissimilarity for all the result set because of the long running time. We choose approximately 250 query results with approximately 150 as the maximum size of result set. The maximum number of gene ontology annotated sequences are 116, 140, and 113 for UniProt, Swiss-Prot and UniRef50 databases, respectively. As seen in the figures, our methods have better results than PSI-BLAST, especially for the beginning instances. According to Wilcoxon signed-rank test, the difference between the proposed diversification methods and BLAST is significant (p-value is smaller than 0.05) for especially the first half of k values. Except for the results of UniProtKB database, the difference between original set and others is significant up to $k \approx 100$. However, for the functional dissimilarity evaluation, we did not find a consistent difference between diversification methods.

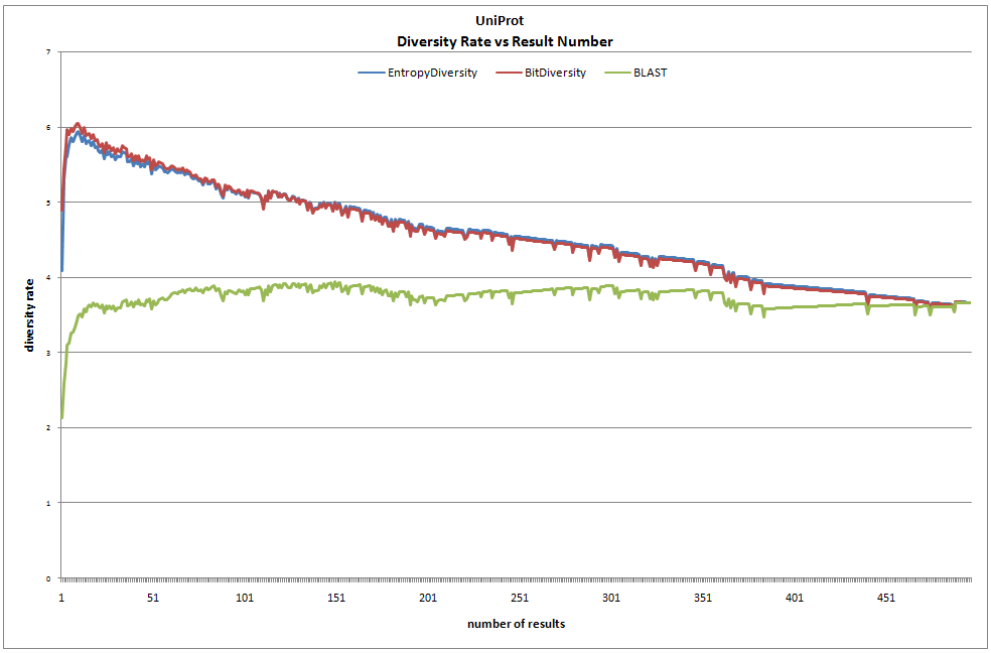


Figure 5.1: Sequence diversity evaluation in UniProtKB database

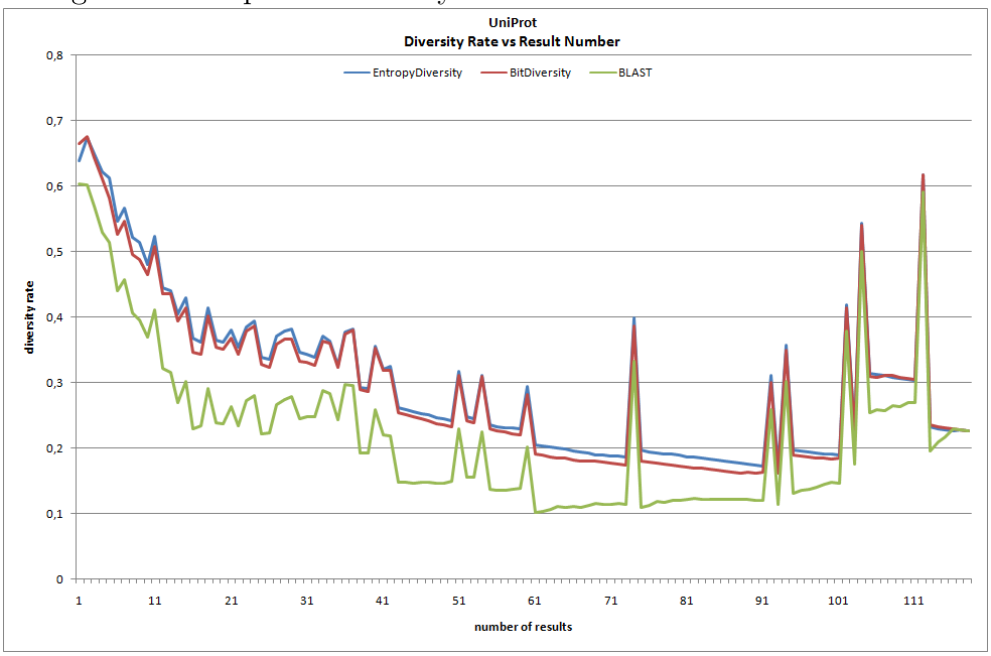


Figure 5.2: Functional dissimilarity evaluation in UniProtKB database

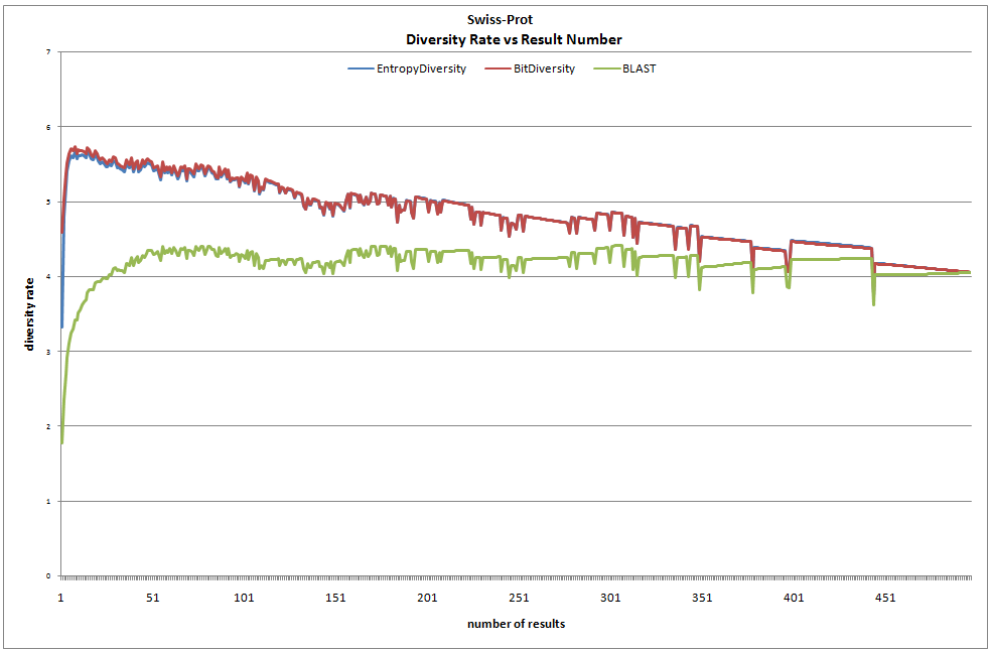


Figure 5.3: Sequence diversity evaluation in Swiss-Prot database

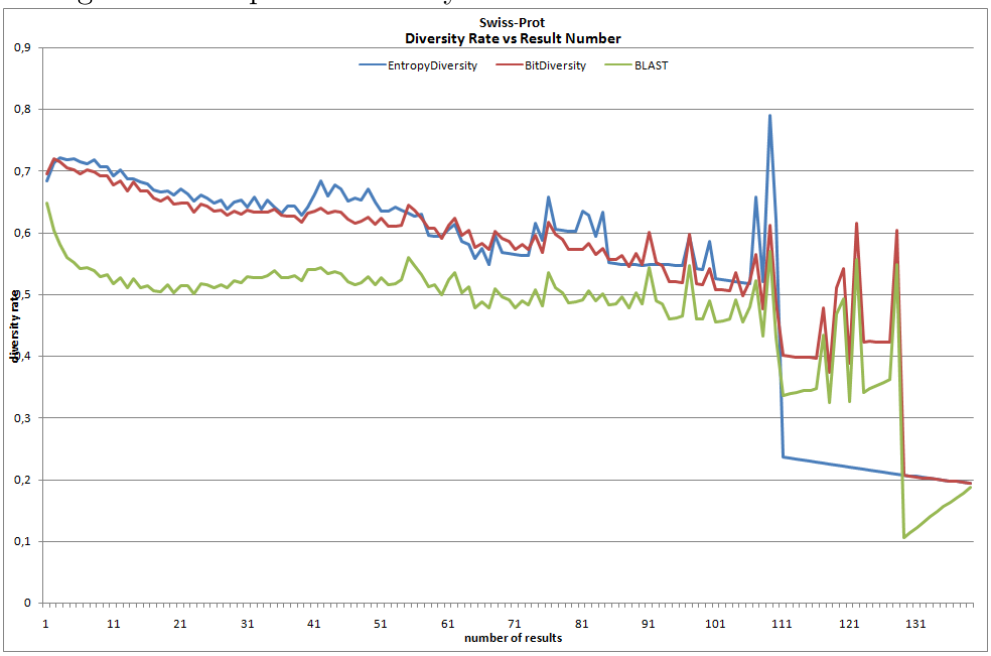


Figure 5.4: Functional dissimilarity evaluation in Swiss-Prot database

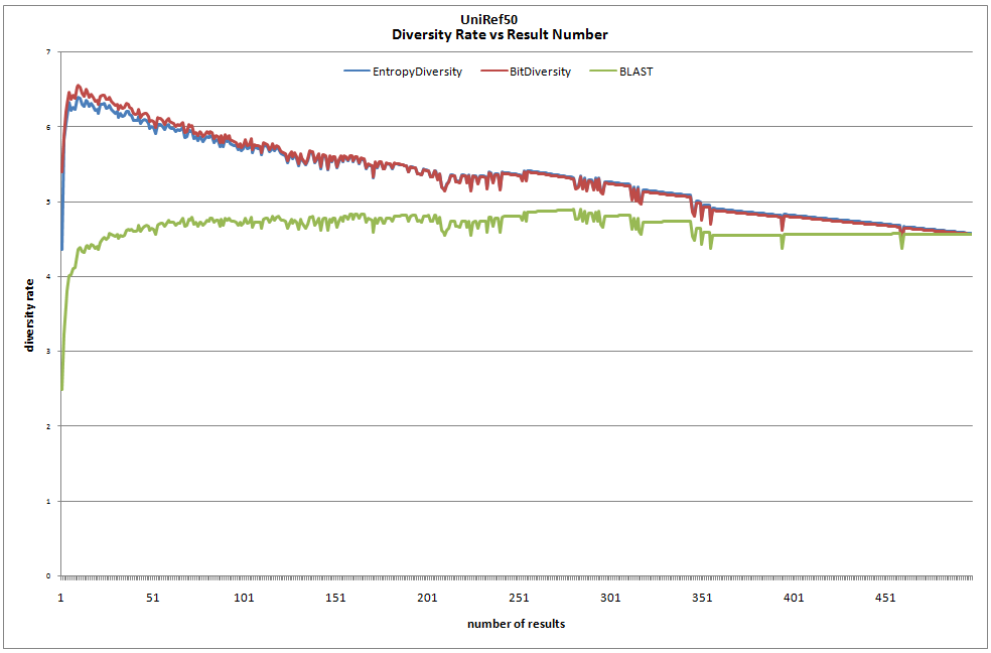


Figure 5.5: Sequence diversity evaluation in UniRef50 database

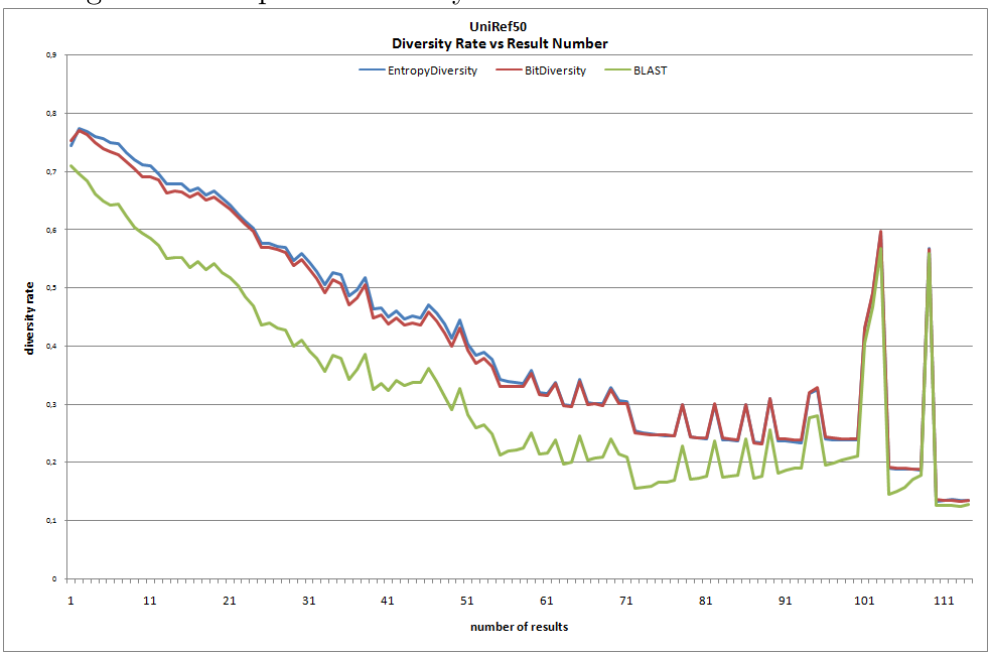


Figure 5.6: Functional dissimilarity evaluation in UniRef50 database

Query coverage of result sequences may be another comparison for result sets. In our experiments, we have full coverage, i.e., every residue of a query is included in one or more result sequences, on 479, 858 and 508 queries out of 1000 random queries in UniProtKB, UniProt/Swiss-Prot and UniRef50 databases, respectively. Since the UniProtKB includes too many sequences, the result sequences may be similar to each other. The UniRef50 consists of much less number of sequences than others; hence, the covered query numbers in these databases are much smaller than in the Swiss-Prot non-redundant database. In UniProtKB, BitDiversity achieves the full coverage with just the 3 percent of result set, while EntDiversity does the same with 4.5 percent and original PSI-BLAST needs 7.5 percent of the result set on the average to reach full coverage. The rates in Swiss-Prot are 1, 1.5, and 4.5 percent. In UniRef50, they are 3, 4, and 10 percent. Note that while investigating coverage, we do not include the first result sequences which are the same sequences as the queries. This may not always be observed; however, in our experiments we use known sequences, and the first result is always the query itself. Figure 5.7, 5.8, 5.9 shows the relation between number of sequences in the result set and query coverage. The figure includes also non-covered query results; the maximum coverage is considered as full coverage for a query. As seen in the figures the diversification methods have more sequences covered in the same percentage of result set. Because the size of result sets for each query may be different, we use the percentage of the result sets. We obtain significant p-values (less than τ=0.05) with Wilcoxon signed-rank test between diversified and original results.

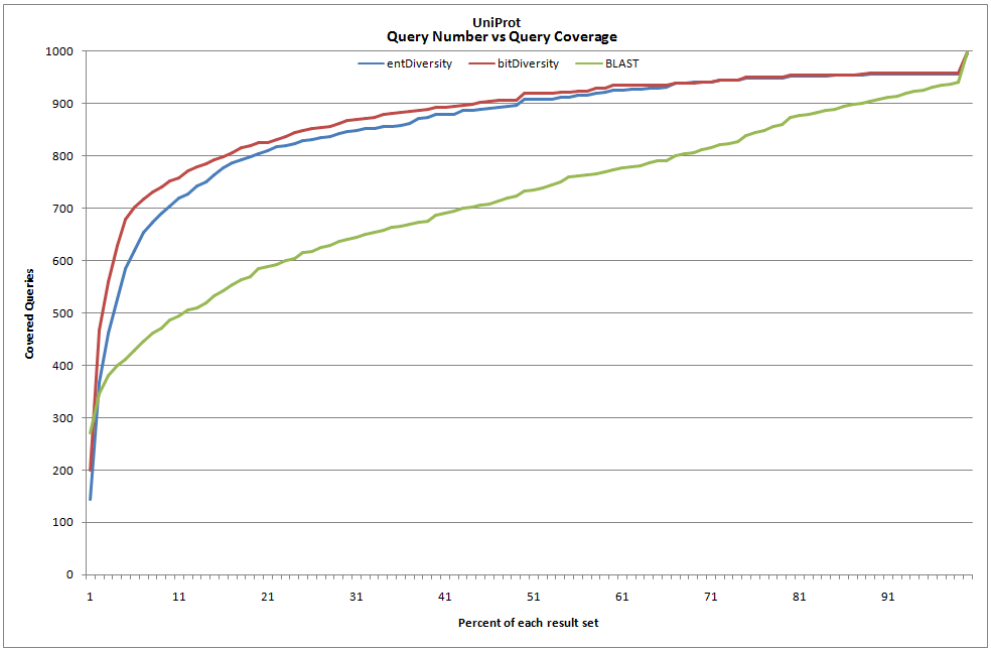


Figure 5.7: Query coverage comparison in UniProtKB database

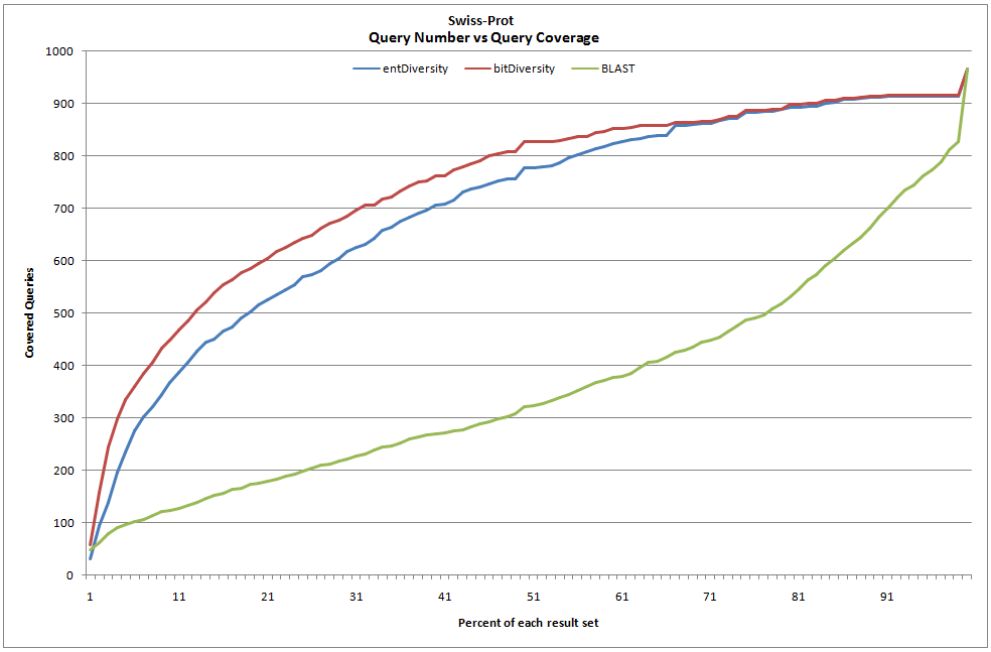


Figure 5.8: Query coverage comparison in Swiss-Prot database

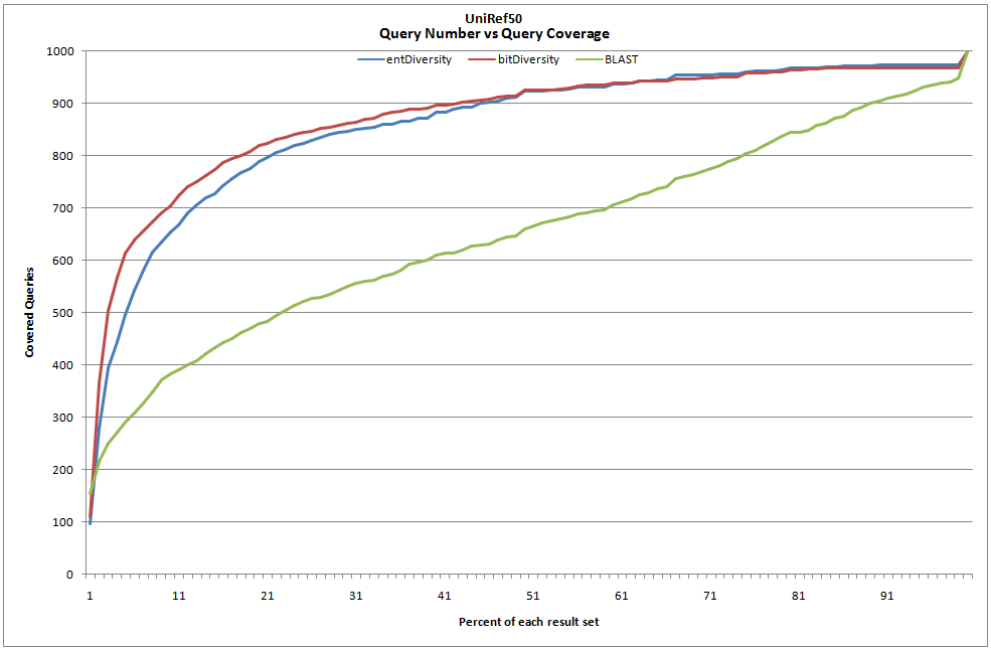


Figure 5.9: Query coverage comparison in UniRef50 database

Chapter 6

Div-BLAST: A Web Based Searching Tool

Div-BLAST is a web based tool that searches primary structure of biological sequences similar to BLAST. Basically, the program queries given sequences in a chosen database and tries to diversify the results by using aforementioned algorithms. Div-BLAST utilizes EBI-EMBL Web Services [33] instead of searching the databases on the server. After web services sends the output of query to the server, the order of initial search set is arranged according to the chosen diversity method mentioned in Chapter 5. In Figure 6.1 the initial screen of the tool is displayed.

6.1 General Overview of Div-BLAST

6.1.1 Input

Sequence: Query sequence is one of required input of the system, naturally. A sequence may be sent via two ways in the screen: One may use the sequence field with FASTA formatted protein sequences or amino acid letter sequences without any format. Another option to query a sequence is to upload a file by

deploying upload function. The content of file must have the same qualification as mentioned for sequence field. The uploaded file must be text, however it can have different extensions e.g. fasta. User cannot continue without entering a sequence. If he wants to get results mistakenly, an error will appear to warn him as seen in Figure 6.2.

The given sequence is saved as a text file named with a random 10 length alphanumeric string. The random string is the request ID which is unique for each session. It will be detailed in "Request ID" section.

E-mail Address: As mentioned in Section 6.2.3 BLAST web services needs e-mail address information as required parameter. Invalid e-mail address formats or empty fields are not allowed. There are warning pop-up windows to prevent mistakes as shown in Figure 6.3.

Query Subrange: This input provides users to give the coordinates of the given query sequence. The tool will apply search to the residues in the range as BLAST does. Because query subrange is a parameter of BLAST tool, Div-BLAST and BLAST have the same characteristic on it.

Request ID: In Div-BLAST, one can reach the search results that he has queried in last three days. On every day, server flushes old search records which stay on the server for the three day period. If user utilize Request ID property, he will confront the exact result page with the given request ID. As said before, the IDs are totally random, not depending on user e-mail or sequence etc.

Database: User has 7 different options for database parameter. UniprotKB is the widest database with redundant and non-redundant protein sequences. It contains UniProtKB/Swiss-Prot and UniProtKB/TrEMBL entries and its total number of sequences is over 30,309,136 for now. Note that the databases are updated and enlarged month by month. Nr database includes all non-redundant protein sequences: GenBank CDS (a coding subsequence of DNA sequence) translations, RefSeq (Reference Sequences) Proteins, PDB (Protein Databank), Swiss-Prot, PIR (Protein Information Resource) and PRF (Protein Research Foundation). It

totally has 31,029,662 protein sequences. Another option for database is Swiss-Prot which is reviewed and annotated proteins of UniProtKB. It is also a non-redundant dataset for proteins. Swiss-Prot contains 539,165 entries. UniRef50, 90 and 100 are UniProtKB reference clusters with different (50%, 90%, and 100%) thresholds. They have 26,071,246, 15,996,810, and 7,939,332 different clusters (reference sequences), respectively.

Maximum Target Sequences: This is the k number for diversity. Maximum target sequence is not employed as a parameter of BLAST search, however it is the number of results that will be diversified. In the phase of BLAST search, any target sequence number is not specified, its default setting is preserved for result number. Div-BLAST does not return more results than given in the input.

BLAST Algorithm: BLAST has more than one algorithms for protein sequence search. Simply, BLASTP compares a protein query to a protein database with basic BLAST algorithm which is mentioned in Section 2.1.1. Position Specific Iterated BLAST (PSI-BLAST) tries to find optimum local alignment by using profiles. These profiles are built by considering evolutionary relationships and using them enables detection of distant relatives of a protein.

Diversity Algorithm: Div-BLAST tool has three options for diversity algorithm. Bitwise comparison and entropy based options diversify the result set with the algorithms. "None" option shows the results with its original BLAST order.

Diversity Percentage vs Similarity: Div-BLAST optimizes similarity and diversity with respect to the given rate by using the input. We adapt the diversity function of maximum marginal relevance [10] represented in Equation 6.1 to add as a feature to Div-BLAST. It refers to *Div* function at Step 9 in Algorithm 1 demonstrated in Chapter 3. The function has two components: one is the difference among diversified set within candidate result sequence and the other is similarity of the sequence to the query. A parameter, λ, determines the contribution percentages of these parts to the total value. As λ increases, the difference becomes more dominant than similarity rate. On the contrary, small λ values make the similarity more significant and eventually when it is 0, the difference

has no importance.

$$diversity = \underset{D_i \in R/S \;\; i \leq k}{\operatorname{argmax}} \; [\lambda difference(D_i, Q, R') + (1 - \lambda) sim(D_i, Q)] \qquad (6.1)$$

To provide the balance between the two aforementioned components, we make them in the same interval: [0,1]. The similarity function is shown in Equation 6.2. For entropy method, the equation is the proportion of the alignment score of aligned fragments with respect to the query, to the maximum alignment score which is calculated by aligning query with query. In bitwise comparison approach, instead of taking into account alignment score value, coverage percent is employed as score. It is because we do not care about amino acid scores in the method, just considered alignment length which is related to coverage percent. The reason of that we do not additionally take the coverage account in entropy-based method is that the alignment score also includes the coverage effect. Lastly, for both of the diversity methods, the difference values are already in the given range [0,1].

$$sim(D_i, Q) = score(D_i, Q)/maxscore(Q) \qquad (6.2)$$

With the help of this optimization to the *Div* function, the effect of diversification is easily observed dynamically in Div-BLAST.

The input information are saved with its settings done by the user for the current session and redirect to waiting screen. Alternatively, one can prefer to see the results in new window. Additionally, the request ID also registered to a server file to control the uniqueness of next IDs.

6.1.2 Progress of Search

As said in the explanation of Div-BLAST, after getting the inputs, an argument list for using web service is built. By looking at the BLAST algorithm, the program decides which web service to be chosen. For BLAST and PSI-BLAST,

there are different web services. First of all, the sequence given in the search page is written to server with its request ID. After all required files are written to the server, web services are utilized by client classes supplied from EBI-EMBL [33] with required libraries. The result of BLAST search is saved in server and by employing file parsing classes, Div-BLAST makes the outputs ready to be diversified. There are xml and text files returning from BLAST as result. Result seqeunces are included in these files. After mining, program builds sequence list with BLAST order.

The waiting screen is illustrated in Figure 6.5. The submission time refers to initial time of search. Status has two different alternatives: "Searching sequences on BLAST" and "Diversifying over BLAST results". After BLAST side is completed, the status is changed and diversification part will be initialized. At the same time, the current time and the time after submission are updated in every second. Elapsed time is given by seconds.

An error page is designed for exceptions occurred on the server side of web services. The page is redirected from wait screen after an error. One of the reasons that user encounters the screen is that the result set of the given query with the given parameters like database, BLAST algorithm, e.g., is empty. It could appear in many situations like giving too short or too long sequences. If BLAST does not see the results significantly similar to the query, it does not return the results. Short alignment mostly does not pass default e-value threshold which is 10 for most of types of BLAST. In addition to the result set problem after search, with wrongly given parameters, or the sequences that include non-amino acid letters. It is hard to detect before sending the arguments to BLAST. It recognizes this kind of problems and informs our server. Also, although we control the pattern of e-mail address before preparing the argument list for web services, wrongly given e-mail addresses with non-existing domain name will cause an exception on the services. Lastly, suddenly occurred internet interruptions and technical troubles on BLAST services are also error factors of web services. The messages of exceptions are directly displayed in the "Error Explanation" section of the page.

6.1.3 Output

After diversification operation, the result page is built with all details of the search as seen in Figure 6.7. Query ID refers to request ID of current search. In the version of Div-BLAST, molecule type of search is amino acid; however in future releases, it is going to be able to query also nucleotides. Query length is the length of searched range of query sequences. It means if range is declared in the query page, query length will not be the same length as the query. The other static properties of search are database, program, and diversity algorithm. There is also a brief explanation for chosen database and diversity method. Diversity percentage is still user-tunable in the result page. As the value of the parameter changes, the related components are renewed: "Graph Summary" and "Descriptions" sections that will be detailed.

In addition to basic information about search, the result of the search is given in two ways. In the first section, "Graph Summary", the alignments of result sequences with respect to query sequence are shown in a colored way. All alignments have a score value and the alignments has the color which is indicated in color key for alignment scores shown at the top of the section. The score of an alignment is calculated by using Smith-Waterman [7] local alignment algorithm. Note that, the scores belong to an alignment not to whole sequence. For example, if a result sequence is aligned to the query in two different locations, the alignment scores are computed seperately. It is because the alignments are independent from each other. Besides, the alignment scores are not normalized with respect to the length of sequences.

To make the alignments more understandable, a scale lies down under color key. The scale has five or six fragments. Fragment number is based on which one is more convenient for sequence length. The scale gives a sight about the length of alignments.

When user hovers a result sequence in "Graph Summary" section, he will see the brief textual description of the sequence. The description generally comprises of the information about the organism where the sequence is found, the type of

the sequences such as protein, primer, mRNA, DNA, e.g. Additionally, when clicking the sequence, all details about alignments of the sequence is presented in "Sequence Detail" section, and the page scrolls down to the detail section. It includes the description mentioned above, total score, e-value, identities, positives and gaps proportions. The whole alignment between subject, i.e., result sequence, and query is shown with the alignment pattern as presented in Figure 6.4. The alignment pattern contains identical residues, positive variations and gaps. Non-aligned parts of query is not seen as gap, it does not get involved in the detail.

In "Descriptions" section, there is a detailed list of result sequences with order, description, score, coverage, e-value and identities information (Figure 6.7). The order refers to the order of sequences after diversification, not original BLAST order. The order attribute could be useful when using sort features. Except description, all features are capable of being sorted. It could be ascending or descending for each one. By clicking the related header, a user can re-order the sequence list in "Descriptions" section. The order feature is used especially to observe the relationship between order of diversified list and other specifications. For example, user can sort the list according to e-value and see the orders of the sequences which have the most amount of e-value.

The section also enables users to download search results. By using download link at the top of the section, user can select or deselect all sequences, download the sequences selected with the current order which does not depend on the order qualification. The download file is a text file includes the basic information about search like database, BLAST algorithm, diversity algorithm etc. and selected sequences alignments with their details.

Similarly to "Graph Summary" section, when user clicks the sequence in the list, "Sequence Detail" section is updated with the information of the sequence and page focuses the detail section.

"Sequence Detail" section is mentioned above frequently. The main specification of the section that it is invisible until user click a sequence in one of the other sections. Additionally, it is not updated until another sequence is selected.

As summarized in this section, Div-BLAST is designed with all user-friendly interface that prevents users giving wrong inputs and shows pop-ups when user wants to get explanation about an input, specification etc. Also, for the next generation of Div-BLAST, a clustering method can be applied to provide users to investigate similar results to a chosen alignment. Hence, users may have the opportunity to see the results that do not appear in top-k diversified list.

6.2 Technologies

6.2.1 ZK: A Java Web Framework

ZK is an Ajax Web application framework that is open source and written in Java [34]. It mainly allows developers to create rich internet applications without knowing Ajax and JavaScript. It is a server-side framework, developers can manipulate inputs and outputs without using client side programming. It also allows users to create HTML or JavaScript objects.

In the phase of programming Div-BLAST, we choose ZK because the main codes of algorithms are written in Java. It is convenient for binding UI components to program algorithm codes.

We use *Eclipse* as development environment with ZK framework plugin. ZK additionally needs a Java web server supporting servlet (2.3 version or later). As servlet container Apache Tomcat 6.0 is employed.

6.2.2 JavaScript

JavaScript (JS) is a popular programming language which is mainly written for web applications. Basically a developer can manipulate HTML objects and client/application interactions with JS. JS does not need any additional environments to run. We use HTML with JS in result page "Graph Summary" section.

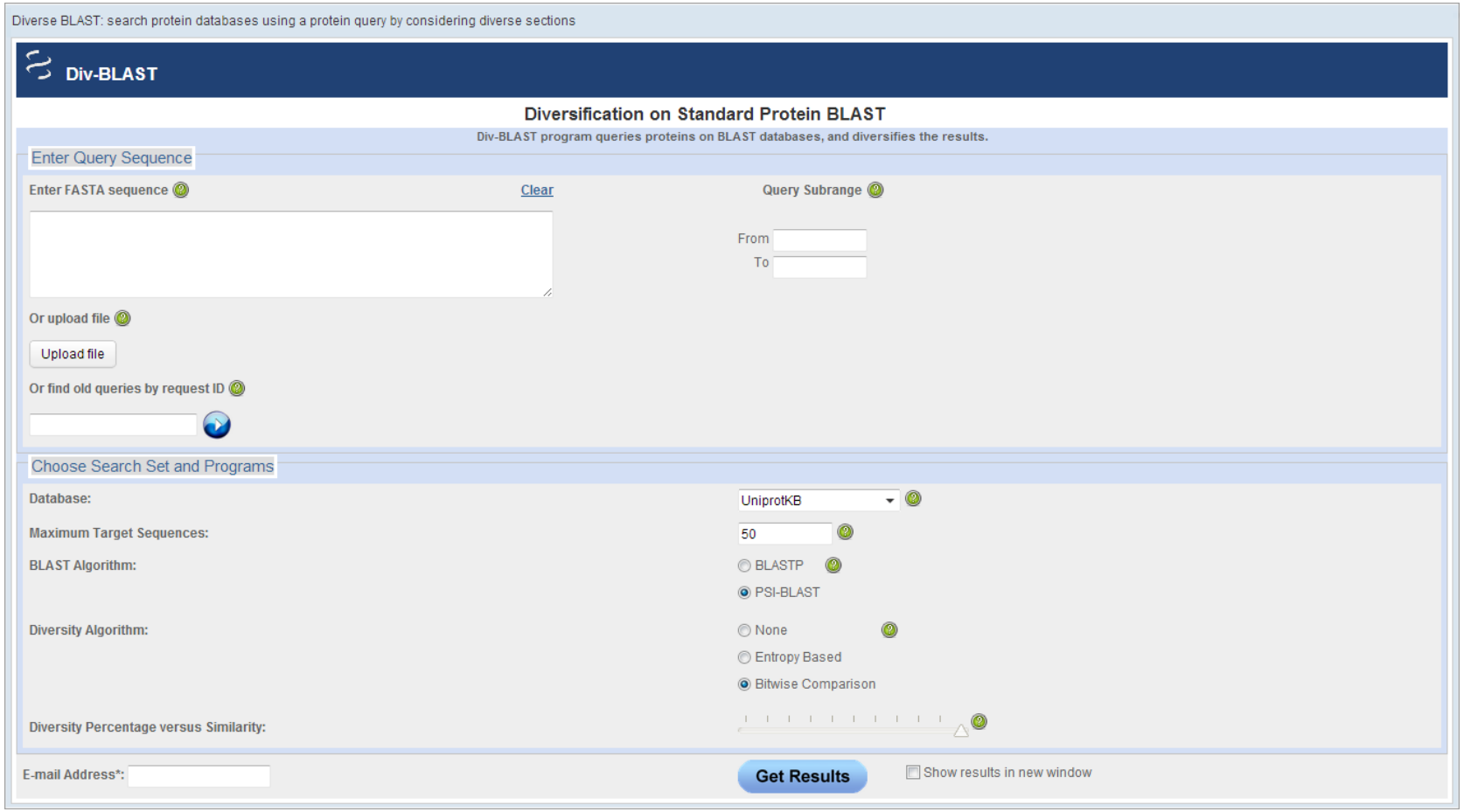


Figure 6.1: Initial screen of diversity tool

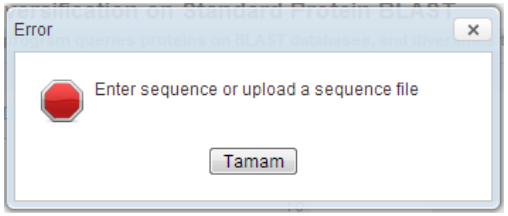


Figure 6.2: Error window warning about sequence input

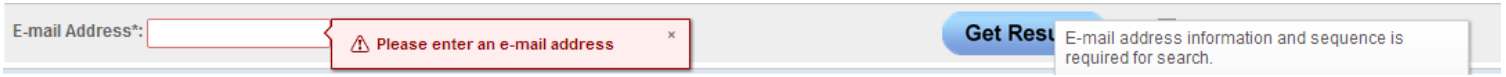


Figure 6.3: Error pop-ups related to e-mail address requirement

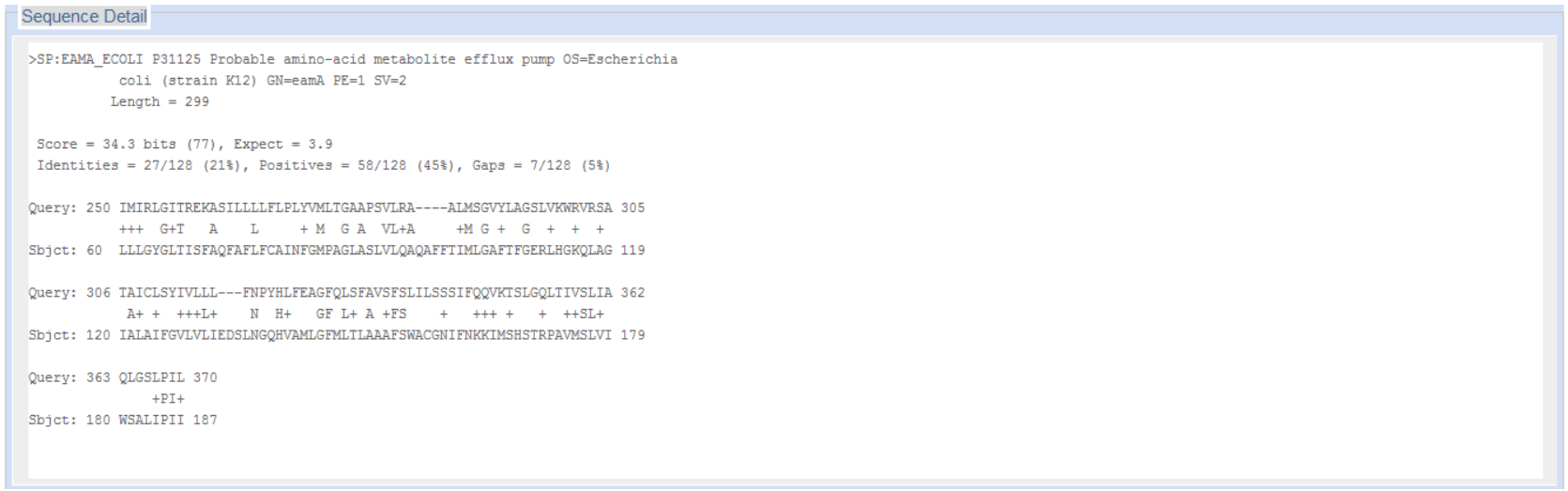


Figure 6.4: Sequence Detail section when a result is chosen

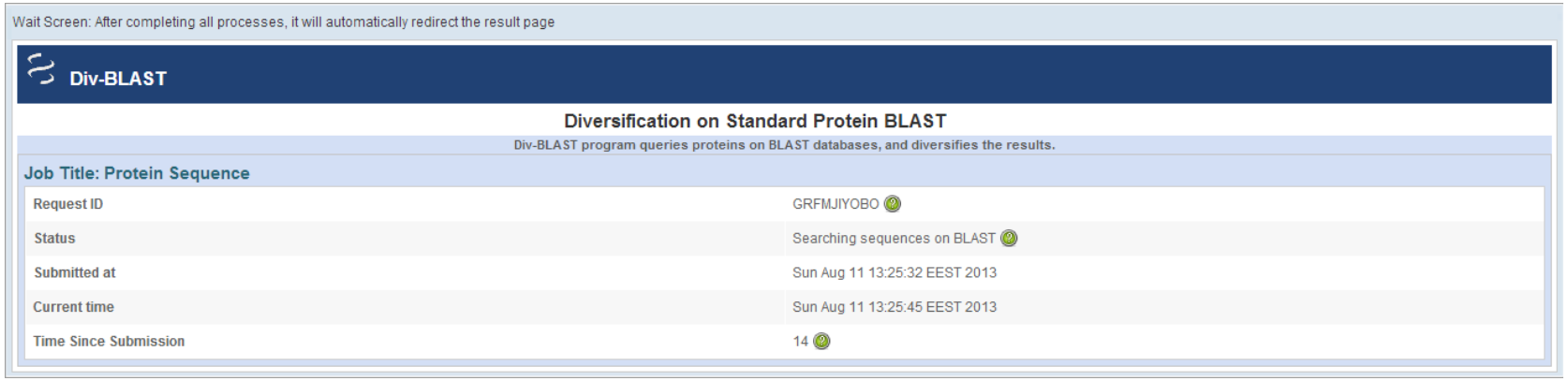
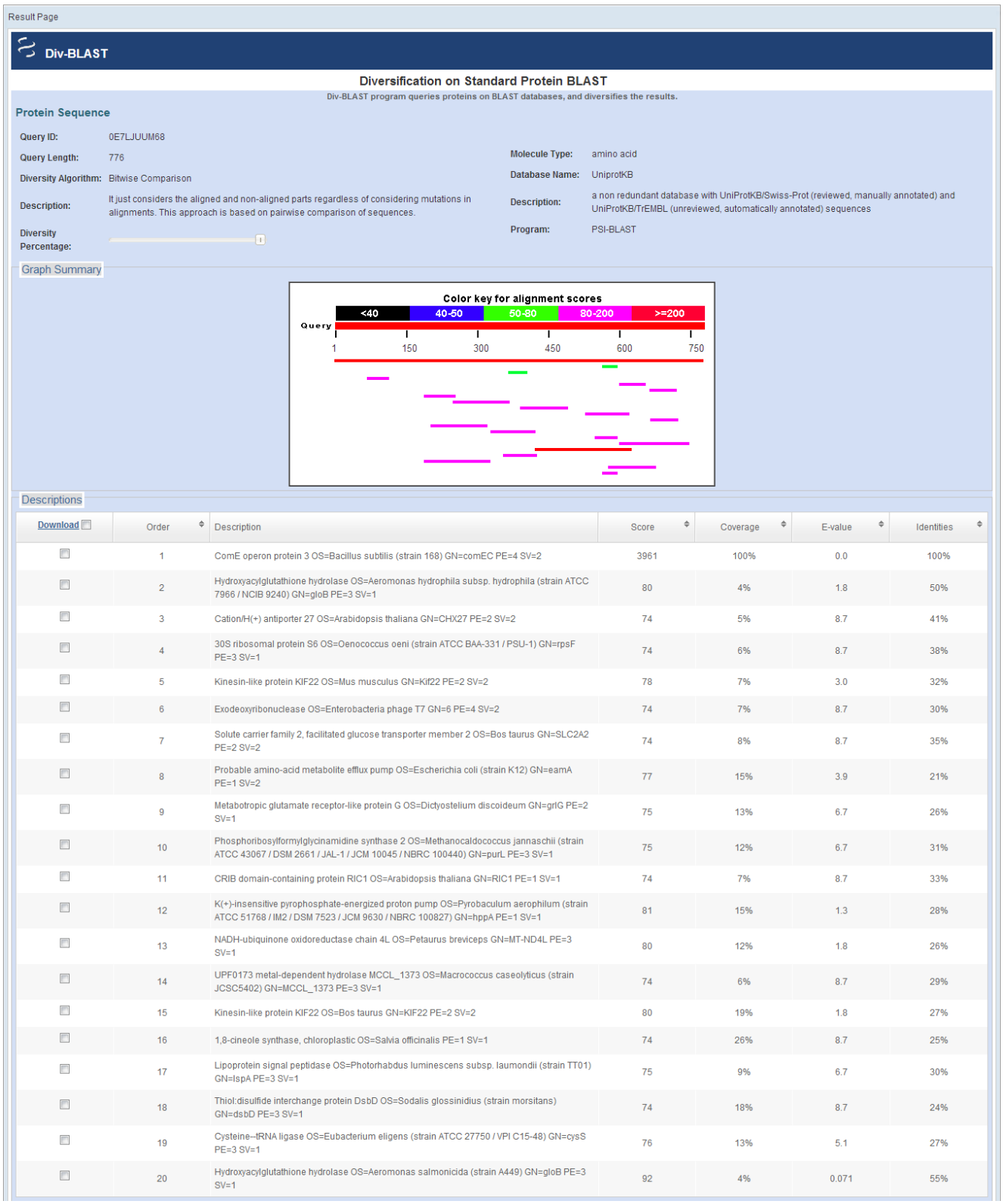


Figure 6.5: Wait Screen

Figure 6.6: Error Screen



Download	Order	Description	Score	Coverage	E-value	Identities
	1	ComE operon protein 3 OS=Bacillus subtilis (strain 168) GN=comEC PE=4 SV=2	3961	100%	0.0	100%
	2	Hydroxyacylglutathione hydrolase OS=Aeromonas hydrophila subsp. hydrophila (strain ATCC 7966 / NCIB 9240) GN=gloB PE=3 SV=1	80	4%	1.8	50%
	3	Cation/H(+) antiporter 27 OS=Arabidopsis thaliana GN=CHX27 PE=2 SV=2	74	5%	8.7	41%
	4	30S ribosomal protein S6 OS=Oenococcus oeni (strain ATCC BAA-331 / PSU-1) GN=rpsF PE=3 SV=1	74	6%	8.7	38%
	5	Kinesin-like protein KIF22 OS=Mus musculus GN=Kif22 PE=2 SV=2	78	7%	3.0	32%
	6	Exodeoxyribonuclease OS=Enterobacteria phage T7 GN=6 PE=4 SV=2	74	7%	8.7	30%
	7	Solute carrier family 2, facilitated glucose transporter member 2 OS=Bos taurus GN=SLC2A2 PE=2 SV=2	74	8%	8.7	35%
	8	Probable amino-acid metabolite efflux pump OS=Escherichia coli (strain K12) GN=eamA PE=1 SV=2	77	15%	3.9	21%
	9	Metabotropic glutamate receptor-like protein G OS=Dictyostelium discoideum GN=grlG PE=2 SV=1	75	13%	6.7	26%
	10	Phosphoribosylformylglycinamidine synthase 2 OS=Methanocaldococcus jannaschii (strain ATCC 43067 / DSM 2661 / JAL-1 / JCM 10045 / NBRC 100440) GN=purL PE=3 SV=1	75	12%	6.7	31%
	11	CRIB domain-containing protein RIC1 OS=Arabidopsis thaliana GN=RIC1 PE=1 SV=1	74	7%	8.7	33%
	12	K(+)-insensitive pyrophosphate-energized proton pump OS=Pyrobaculum aerophilum (strain ATCC 51768 / IM2 / DSM 7523 / JCM 9630 / NBRC 100827) GN=hppA PE=1 SV=1	81	15%	1.3	28%
	13	NADH-ubiquinone oxidoreductase chain 4L OS=Petaurus breviceps GN=MT-ND4L PE=3 SV=1	80	12%	1.8	26%
	14	UPF0173 metal-dependent hydrolase MCCL_1373 OS=Macrococcus caseolyticus (strain JCSC5402) GN=MCCL_1373 PE=3 SV=1	74	6%	8.7	29%
	15	Kinesin-like protein KIF22 OS=Bos taurus GN=KIF22 PE=2 SV=2	80	19%	1.8	27%
	16	1,8-cineole synthase, chloroplastic OS=Salvia officinalis PE=1 SV=1	74	26%	8.7	25%
	17	Lipoprotein signal peptidase OS=Photorhabdus luminescens subsp. laumondii (strain TT01) GN=lspA PE=3 SV=1	75	9%	6.7	30%
	18	Thiol:disulfide interchange protein DsbD OS=Sodalis glossinidius (strain morsitans) GN=dsbD PE=3 SV=1	74	18%	8.7	24%
	19	Cysteine--tRNA ligase OS=Eubacterium eligens (strain ATCC 27750 / VPI C15-48) GN=cysS PE=3 SV=1	76	13%	5.1	27%
	20	Hydroxyacylglutathione hydrolase OS=Aeromonas salmonicida (strain A449) GN=gloB PE=3 SV=1	92	4%	0.071	55%

Figure 6.7: Result screen of Div-BLAST

It is more permissive than ZK in configuring colored alignment table and does not put additional spans as in ZK.

6.2.3 EMBL-EBI Web Services

While many bioinformatics tools can be used over web, they usually just provide web interfaces which are designed for users, directly. When researching about an issue and experience on many samples, it is impossible to utilize this kind of programs which have no systematic access to a resource, software or data. EMBL-EBI supplies web services which are accessible via program languages such as C# or Java [33] [35] . There are many web services served by EMBL-EBI, in Div-BLAST NCBI BLAST and PSI-BLAST services are deployed. In both, there are required arguments for some parameters such as e-mail address. Here are some of the web services' parameters which may be used in Div-BLAST search [35]:

Parameter	**Description**
email	email address retrieved from email input of search page.
program	valid for NCBI BLAST Web Service with BLASTP value (in later versions nucleotide search may be activated and the parameter value may change like BLASTN, BLASTX..)
database	could be uniprotkb, uniprot_swissprot, nr,...
stype	protein(other option is nucleotide)
seqrange	(optional) x-y from x to y including both of them
outfile	in Div-BLAST it is given request ID.
sequence	the file address of sequence. The sequence given in search page is saved with its request ID and its location is used as argument.

Table 6.1: Parameters and descriptions that belong to BLAST Web Services and used by Div-BLAST

While using web services, we use client classes written in Java with a little alteration. After web service writes the output files of search to our server, we mine the results for diversification.

To sum up the Div-BLAST tool:

- In the first page the parameters related to both search in BLAST such as database, program etc. and diversification such as diversity rate, k number.
- An argument list is created for BLAST and via web services the query is submitted.
- While the all processes are done both querying and diversifying a waiting screen is shown on the screen.
- The result page is built with the results without any exceptions.
- If any problem occurs on any operation, it redirects error page with exception explanation.

Chapter 7

Conclusion

Diverse browsing of sequences and structures is essential for exploratory research in bioinformatics. The current approach of curating the databases and eliminating identical sequences or fragments is costly and prone to error. In addition, most queries still contain results with too much redundancy, as we illustrated in the experimental section. Alignment and search tools need to perform diversification tailored for each query. To the best of our knowledge, this is the first work that investigates diversity in sequence search and alignment. We proposed quality measures and methods to diversify the results of sequence similarity search tools. As the result set already includes top matching sequences, we focused on selecting a diverse subset of this result. To obtain diverse results, one could specify a similarity threshold and omit the sequences that have more similarity than the threshold. However, this approach would fail to return enough number of results and may not supply the desired diversity regarding the query. To overcome this problem, we first presented a pairwise bit comparison approach, BitDiversity, by treating the sequence matches as bit sequences. BitDiversity stresses the diversity in matching locations without considering amino acid differences in those locations. Diversity rate is calculated with the XOR operation for the bit sequences of two sequences. We proposed another approach based on an entropy-based diversity algorithm and focused on the diversification at the amino acid level, as well. In EntropyDiversity, we compute the entropy of each i^{th} position, it is from

1 to the size of the multiple alignment of all result sequences with respect to the query, and in each iteration the sequence which maximizes entropy as chosen to be added to the result set. For both the proposed approaches, we investigated design alternatives for calculating the difference between two sequences, and chose the appropriate ones. To evaluate the sequence based diversity, we developed a new algorithm based on Rao's quadratic entropy providing an entropy measure by considering the distances. Our methods significantly outperformed the original result sets on various databases including the non-redundant ones. We also evaluated the functional diversity of the result set based on the GO terms. Our methods improved the original result set also in terms of functional diversification.

Diversification of sequence similarity search results promises biologists more efficient exploration of the potential functional landscape of the query sequence. By integrating other biological information such as subcellular localization and related pathways to the diversity measures, diversification may be tailored to specific biological goals. We built a web based sequence search and diversity tool named as Div-BLAST. The software provides users to search their queries on BLAST databases and as depending on the choice, the output is diversified with the chosen aforementioned algorithms. In Div-BLAST, one can adjust settings of BLAST and diversification search, and after getting the results he can calibrate the diversity rate versus similarity rate, notice the alteration of the result set order dynamically, re-rank the result sequences with respect to sequence other features such as score, coverage etc., save the results of his search. Moreover, user can reach old searches without downloading the query results by employing the given ID.

There are several directions one can follow to extend the proposed methods. For example, the performance can be improved by controlling the query coverage on every possible residue. As the proposed approach already outperforms BLAST on query coverage; an explicit query coverage based diversity design may further enhance the performance. Another future work is to highlight the conserved regions in the result set. While presenting the diverse results, one can focus on the observed sections in every result sequence in a result set with respect to the query. Conservation is an important concept in proteins and highlighting the

parts and diversifying the results also by considering these regions may increase the quality of diversification, biologically. Another research problem is to evaluate the significance of the local matches in the diverse results. The currently used e-value is affected by the length of the query and whole result sequence, i.e., not only the aligned sections. In diversification, we only take the aligned fragments into consideration and a new significant value may be more illustrative for diversified results. Finally, Div-BLAST software can be extended for nucleotide sequences, e.g., DNA and mRNA. Additional parameters for search may be integrated for search such as e-value threshold, similarity matrix choice, etc.

Bibliography

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[2] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped blast and psiblast: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[3] W. Pearson and D. Lipman, "Improved tools for biological sequence comparison," in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 85, pp. 2444–2448, 1988.

[4] R. L. Santos, C. Macdonald, and I. Ounis, "Exploiting query reformulations for web search result diversification," in *Proceedings of the 19th International Conference on World Wide Web*, pp. 881–890, ACM, 2010.

[5] D. W. Mount, *Bioinformatics: sequence and genome analysis*, ch. Phylogenetic Prediction, pp. 53–134. CSHL Press, 2004.

[6] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.

[7] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195–197, 1981.

[8] S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," in *Proceedings of the National Academy of Sciences*, vol. 89, pp. 10915–10919, National Acad Sciences, 1992.

[9] E. G. Shpaer, M. Robinson, D. Yee, J. D. Candlin, R. Mines, and T. Hunkapiller, "Sensitivity and selectivity in protein similarity searches: A comparison of smith waterman in hardware to blast and fasta," *Genomics*, vol. 38, no. 2, pp. 179 – 191, 1996.

[10] J. Carbonell and J. Goldstein, "The use of mmr, diversity-based reranking for reordering documents and producing summaries," *In Proc. of SIGIR*, pp. 335–336, 1998.

[11] A. Jain, P. Sarda, and J. R. Haritsa, "Providing diversity in k-nearest neighbor query results," in *Advances in Knowledge Discovery and Data Mining*, pp. 404–413, Springer, 2004.

[12] J. Haritsa, "The kndn problem: A quest for unity in diversity.," *IEEE Data Engineering Bulletin*, vol. 32, pp. 15–22, 2009.

[13] H. Chen and D. R. Karger, "Less is more: probabilistic models for retrieving fewer relevant documents," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 429–436, ACM, 2006.

[14] C. Yu, L. Lakshmanan, and S. Amer-Yahia, "It takes variety to make a world: diversification in recommender systems," in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 368–378, ACM, 2009.

[15] C. Yu, L. Lakshmanan, and S. Amer-Yahia, "Recommendation diversification using explanations," in *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pp. 1299–1302, IEEE, 2009.

[16] B. Liu and H. V. Jagadish, "Using trees to depict a forest," *PVLDB*, vol. 2, pp. 133–144, 2009.

[17] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. A. Yahia, "Efficient computation of diverse query results," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pp. 228–236, IEEE, 2008.

[18] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon, "Novelty and diversity in information retrieval evaluation," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 659–666, ACM, 2008.

[19] H. T. Dang, D. Kelly, and J. J. Lin, "Overview of the trec 2007 question answering track," in *TREC*, vol. 7, p. 63, Citeseer, 2007.

[20] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong, "Diversifying search results," in *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pp. 5–14, ACM, 2009.

[21] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.

[22] L. Jost, "Entropy and diversity," *Oikos*, vol. 113, no. 2, pp. 363–375, 2006.

[23] C. Sander and R. Schneider, "Database of homology-derived protein structures and the structural meaning of sequence alignment," *Proteins: Structure, Function, and Bioinformatics*, vol. 9, no. 1, pp. 56–68, 1991.

[24] C. R. Rao, "Diversity and dissimilarity coefficients: a unified approach," *Theoretical Population Biology*, vol. 21, no. 1, pp. 24–43, 1982.

[25] S. Pavoine, S. Ollier, and D. Pontier, "Measuring diversity from dissimilarities with rao's quadratic entropy: are any dissimilarities suitable?," *Theoretical Population Biology*, vol. 67, no. 4, pp. 231–239, 2005.

[26] D. E. Almonacid and P. C. Babbitt, "Toward mechanistic classification of enzyme functions," *Current Opinion in Chemical Biology*, vol. 15, no. 3, pp. 435–442, 2011.

[27] M. V. Omelchenko, M. Y. Galperin, Y. I. Wolf, and E. V. Koonin, "Research non-homologous isofunctional enzymes: A systematic analysis of alternative solutions in enzyme evolution," *Biology Direct*, vol. 5, p. 31, 2010.

[28] C. Pesquita, D. Faria, A. O. Falcao, P. Lord, and F. M. Couto, "Semantic similarity in biomedical ontologies," *PLoS Computational Biology*, vol. 5, no. 7, 2009.

[29] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, and S. S. Dwight, "Gene ontology: tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.

[30] J. Z. Wang, Z. Du, R. Payattakool, S. Y. Philip, and C.-F. Chen, "A new method to measure the semantic similarity of go terms," *Bioinformatics*, vol. 23, no. 10, pp. 1274–1281, 2007.

[31] D. A. Wolfe and M. Hollander, *Nonparametric Statistical Methods*, pp. 51 – 56. John Wiley New York, 1973.

[32] B. E. Suzek, H. Huang, P. McGarvey, R. Mazumder, and C. H. Wu, "Uniref: comprehensive and non-redundant uniprot reference clusters," *Bioinformatics*, vol. 23, no. 10, pp. 1282–1288, 2007.

[33] `http://www.ebi.ac.uk/Tools/webservices/services/sss/ncbi_blast_soap`.

[34] M. Stäuble and H.-J. Schumacher, *ZK Developer's Guide: Developing responsive user interfaces for web applications using Ajax, XUL, and the open source ZK rich web client development framework*. Packt Publishing, 2008.

[35] P. May, H.-C. Ehrlich, and T. Steinke, "Zib structure prediction pipeline: Composing a complex biological workflow through web services," in *Euro-Par 2006 Parallel Processing*, pp. 1148–1158, Springer, 2006.